Applications using GenAVB/TSN stack for Industrial Automation

Maanasa Sachidanand, NXP Semiconductors

maanasa.sachidanand@nxp.com

About Me

Embedded Software Engineer at NXP Semiconductors, France since 2022

Working on GenAVB/TSN stack in the Real Time Networking team

Previously, Digital Design Engineer at Texas Instruments, India

Outside Work:

Founder of WomenAdoreTech

Children Volunteer

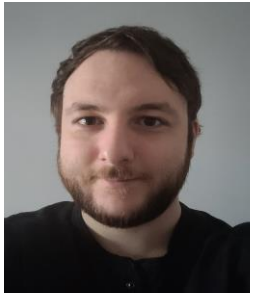
GenAVB/TSN Development Team

Antoine Boiteau

Rui Sousa

Laurent Brando

Marouen Ghodhbane



Alexandre Howard

Maanasa Sachidanand

Tobias Aguiar Silva De Almeida

Aziz Sellami

Agenda

- GenAVB/TSN Stack
- GenAVB/TSN stack Integration
- Industrial applications
- Challenges

01

GenAVB/TSN Stack

Brief Outline

AVB and TSN Standards

Audio Video Bridging (AVB)

- Ethernet extension defined by IEEE to provide guaranteed quality of service.
- Helps create a single network for audio, video, and other data like control information, using AVB-compatible switches.

Time-sensitive networking (TSN)

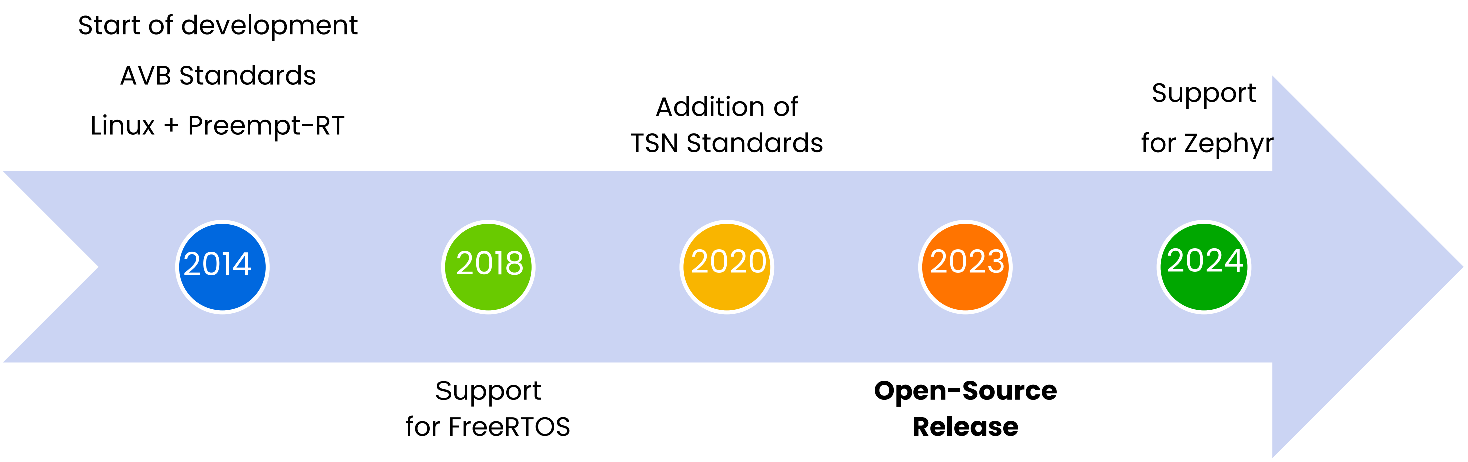
- Ethernet extension on top of AVB standards
- Makes Ethernet-based networks more deterministic.
- Serves highly-critical and time-sensitive streams

Standards provide improved synchronization, Interoperability, bounded low latency, low jitter and reliability for switched Ethernet networks.

What is the GenAVB/TSN stack?

- A generic Audio Video Bridging (AVB) and Time Sensitive Networking (TSN) software stack developed by NXP Semiconductors.
- Supports NXP MCUs and MPUs
- Runs on several OS: FreeRTOS, Zephyr and Linux
- Runs on multiple ARM CPU architectures:
  - ARMv7-M, ARMv7-A, ARMv8-M, ARMv8-A
- Runs on several ARM Embedded SoC cores:
  - ARM Cortex M33, M7, A7, A9, A35, A53, A55, A72
- OS/Hardware independent core code, greatly reducing re-development cost
- Scalable architecture

GenAVB/TSN Timeline



Features of the GenAVB/TSN Stack

- IEEE-802.1AS-2020 (gPTP) implementation, both time-aware Bridge and Endpoint support.
- IEEE 802.1CB-2017 implementation for Frame Replication and Elimination for Reliability.
- IEEE-802.1Q-2022 implementation, both Bridge and Endpoint Support
    - VLAN/FDB
    - Stream Reservation Protocol (Qat-2010)
    - Scheduled Traffic (Qbv-2015)
    - Frame preemption (Qbu-2016)
    - Per Stream Filtering and Policing (Qci-2017)
    - Forwarding and Queuing for Time-Sensitive Streams (Qav-2009)

Features of the GenAVB/TSN Stack

- IEEE 802.3br-2016 implementation for Interspersing Express Traffic.
- IEEE-1722-2016 implementation, with MAAP support.
- IEEE-1722.1-2013 implementation, with support for Milan 1.1a mode.
- IEC 62439-3:2022
    - High-availability Seamless Redundancy (HSR)
- Protocol stacks running in standalone userspace processes for Linux, and dedicated threads for FreeRTOS/Zephyr.
- Multiple reference applications illustrating AVB/TSN use cases.

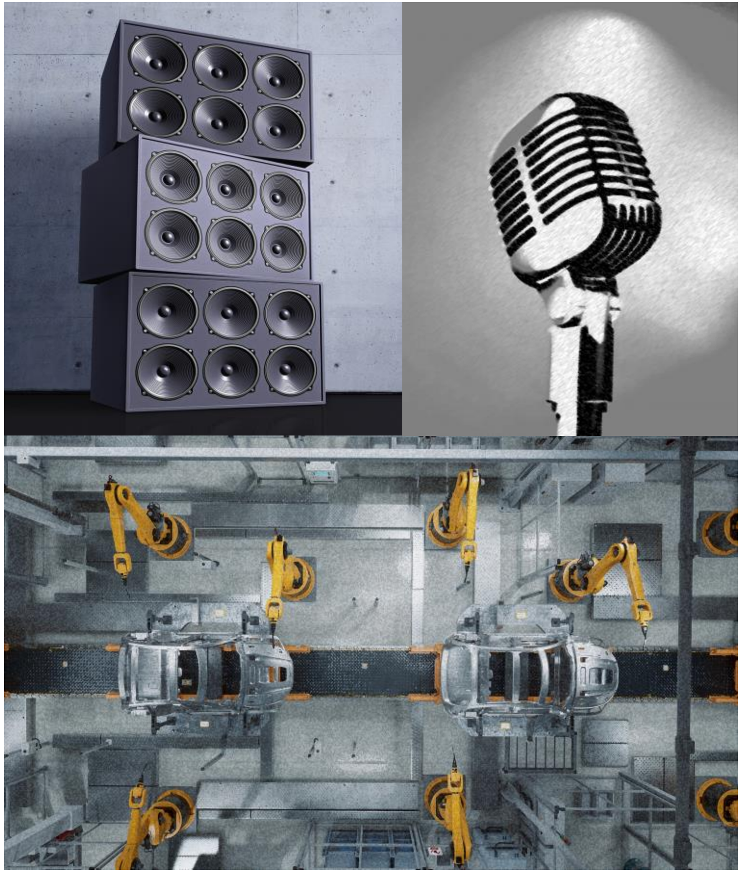
Note: The software stack relies on TSN functionality implemented by the Eth MAC/switch hardware controllers

Targeted Applications

- Industrial, Consumer and Automotive

- AVB technology focused on Audio/Video streaming applications (Speaker, Microphone, Media server, …)

- TSN technology focused on Industrial endpoint (Servo drive, Remote IO, PLC, …) with requirement of better performance and reliability.



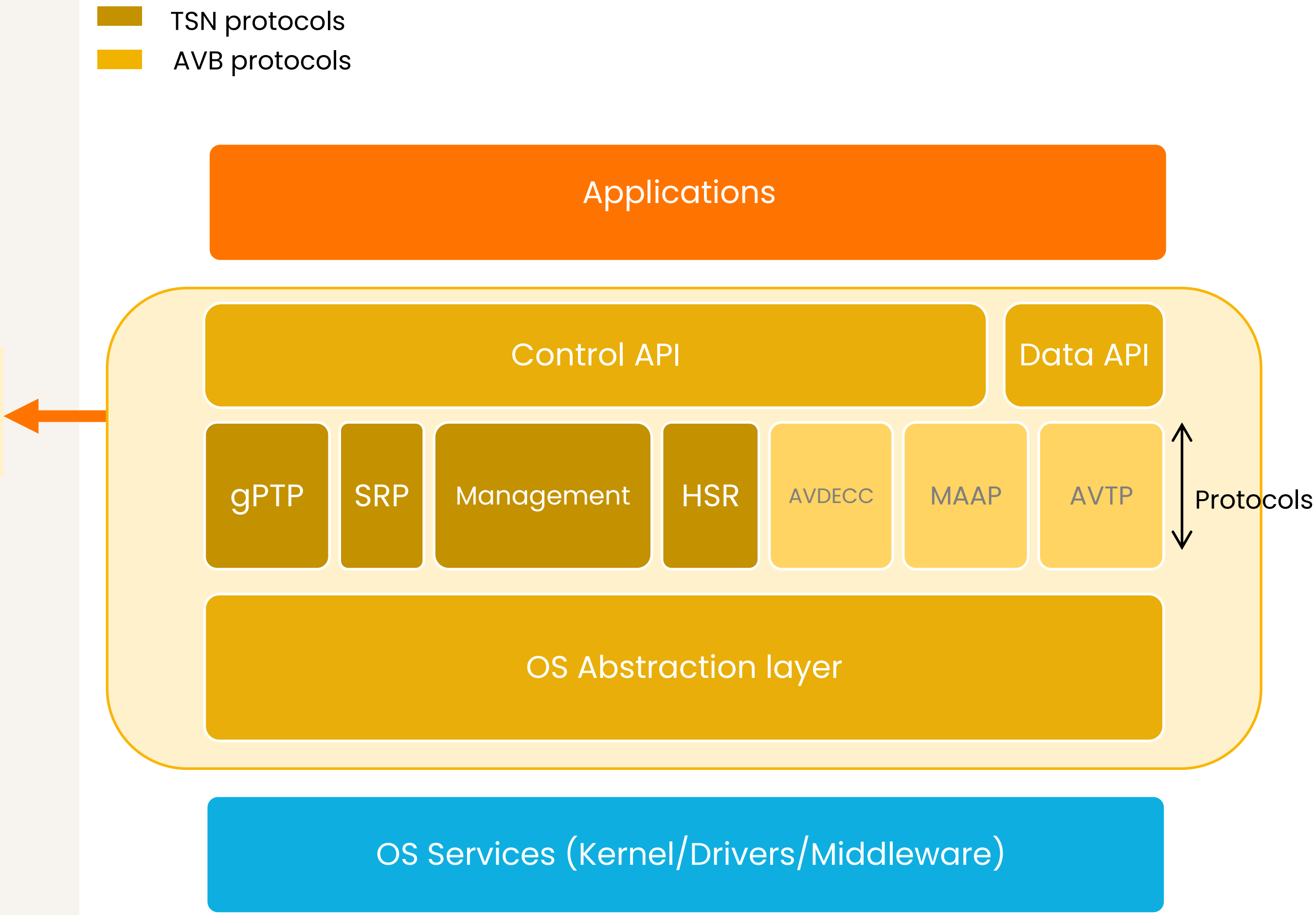
02

GenAVB/TSN Stack Integration

Software Architecture

Software Architecture

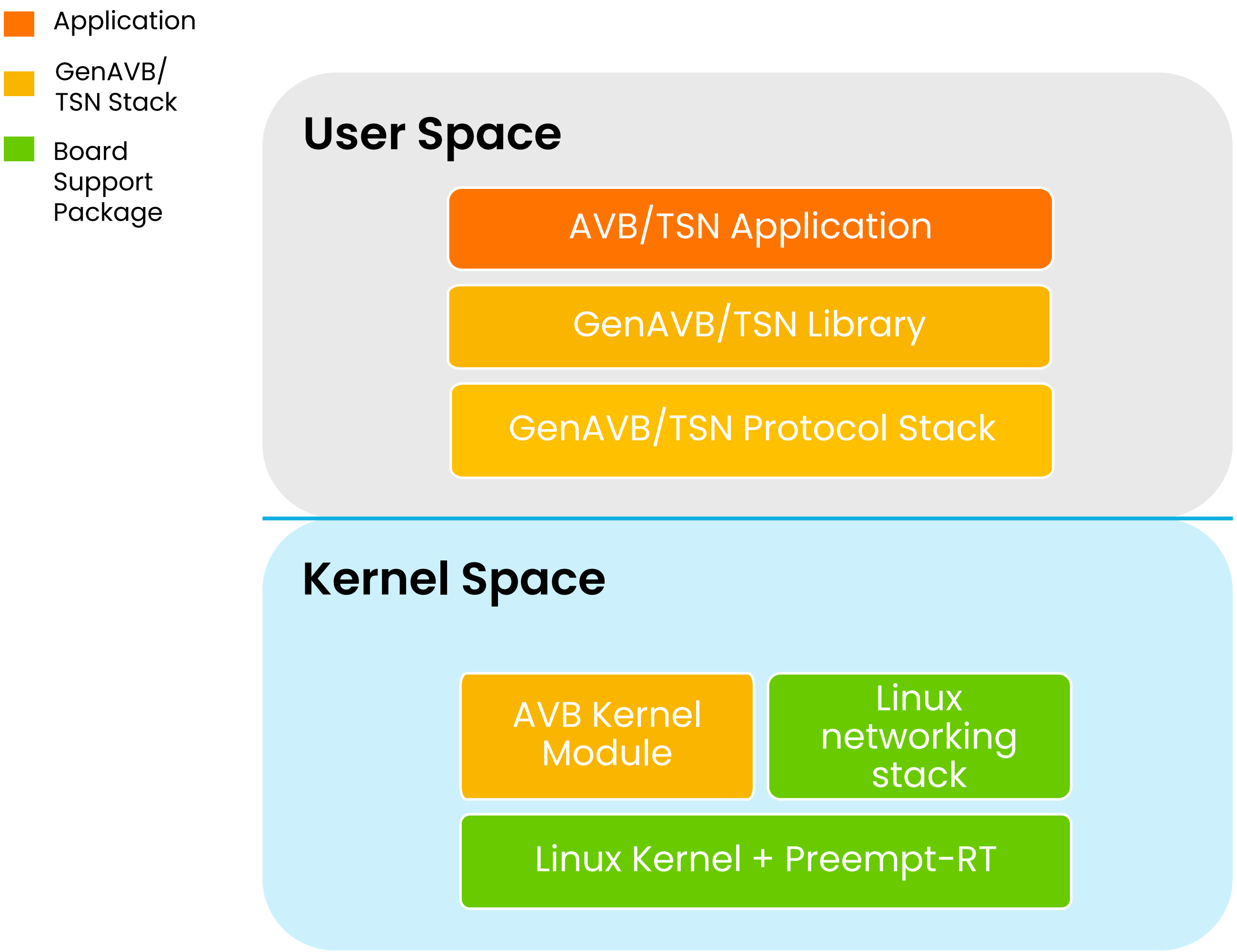
GenAVB/TSN Stack



Software Architecture

GenAVB/TSN Stack in Linux OS

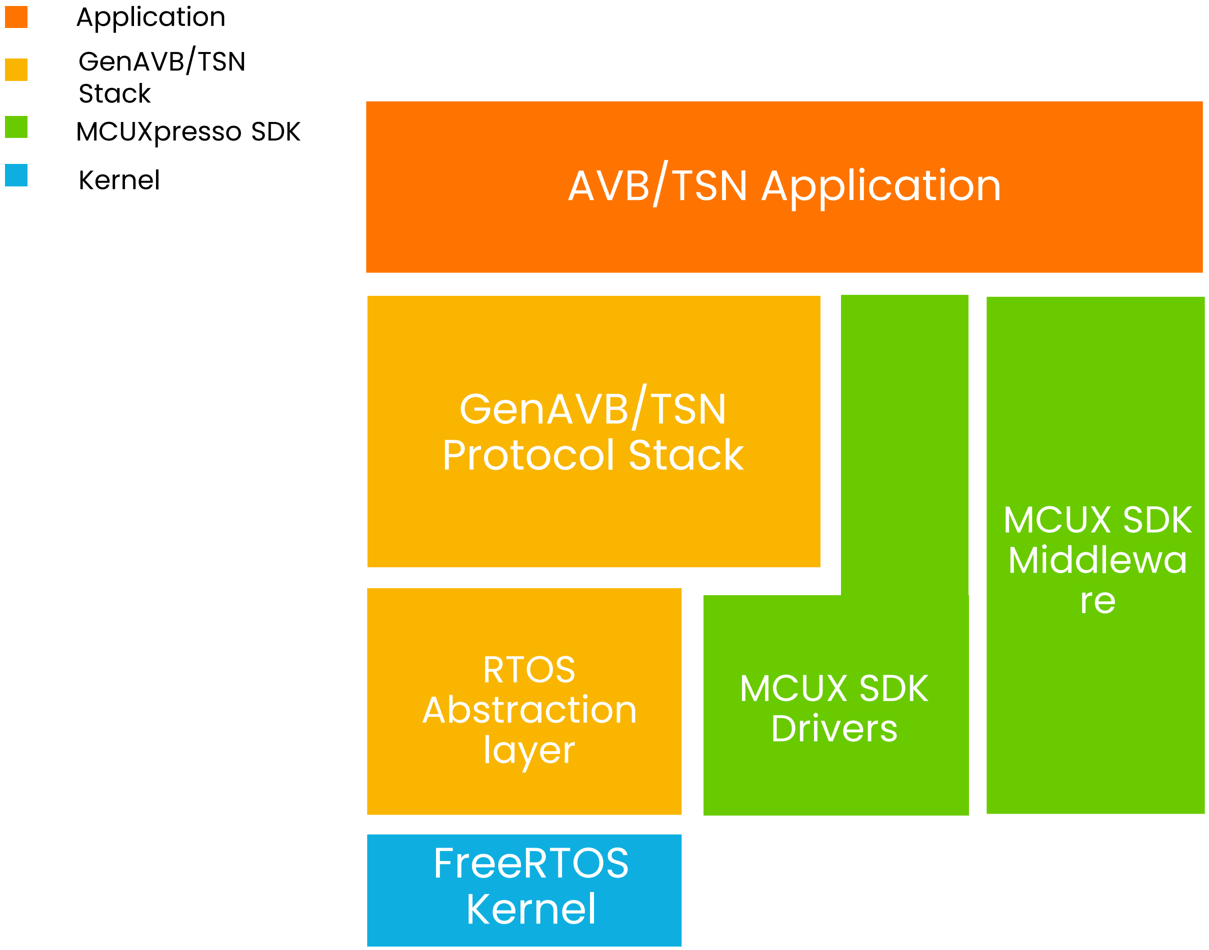
- Distributed as part of NXP Real Time Edge Yocto
- BSD-3 + GPLv2 license
- Runs on Cortex-A
- AVB/TSN applications link with libgenavb
- Supported devices: i.MX 8M Mini, i.MX 8M Plus, i.MX 8XLite, i.MX 93 and LS1028A



Software Architecture

GenAVB/TSN Stack in FreeRTOS OS

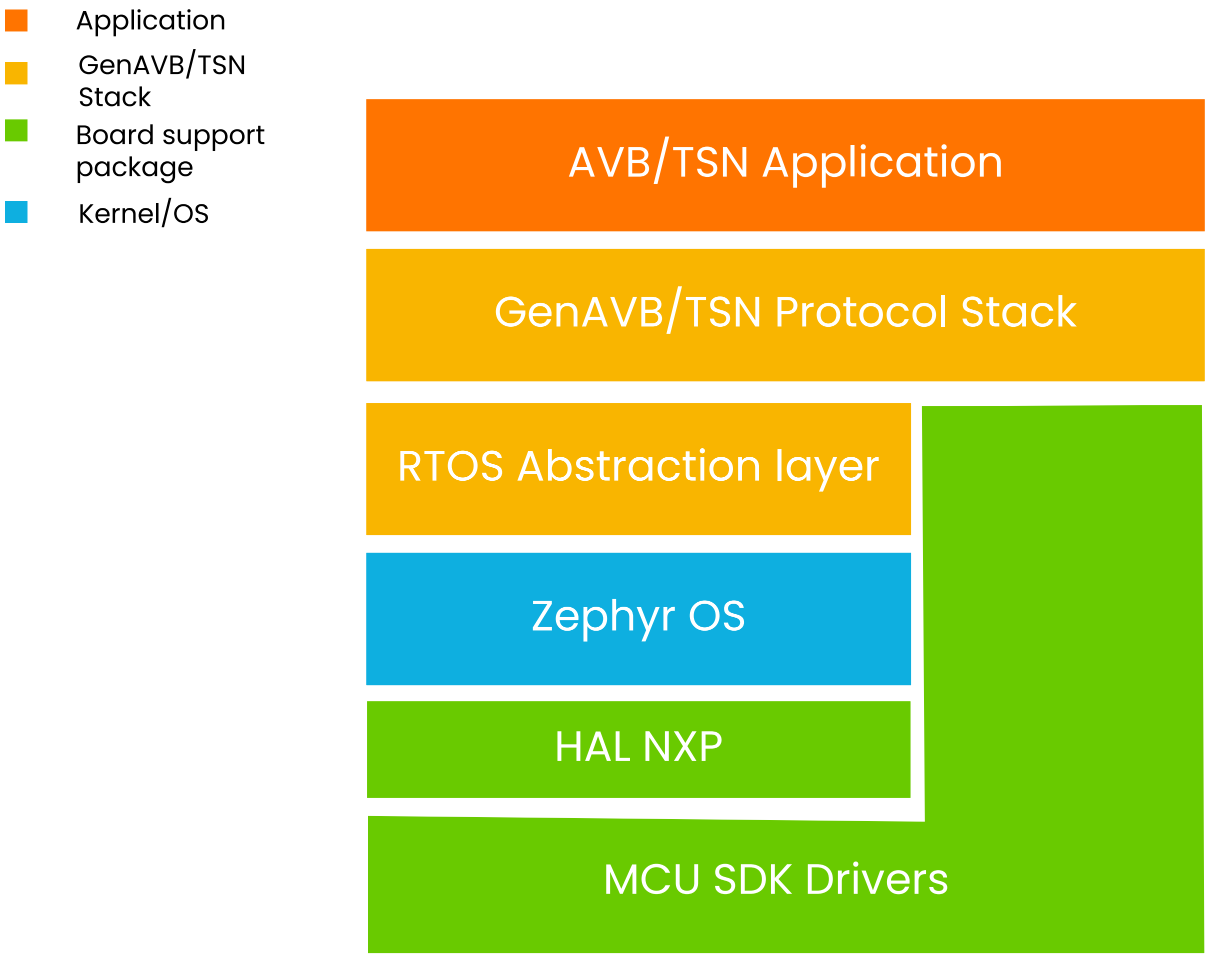
- Distributed with NXP MCUXpresso SDK
- BSD-3 License
- Runs on Cortex-M
- Supported devices: i.MX RT1050, i.MX RT1170 and i.MX RT1180



Software Architecture

GenAVB/TSN Stack in Zephyr OS

- Distributed with NXP Real Time Edge Yocto
- BSD-3 License
- Runs on Cortex-A (under hypervisor)
- Supported devices: i.MX 8M Mini, i.MX 8M Plus, i.MX 93



Where can you find the stack?

- All code released in source form under an Open-Source License
- Released in GitHub
  https://github.com/NXP/GenAVB_TSN

MCU FreeRTOS applications will be available soon on GitHub

NXP SoC enablement software distribution:

- Real-Time Edge Software for MPUs (Linux, Zephyr): https://www.nxp.com
- MCUXpresso for MCUs (FreeRTOS): online download on https://mcuxpresso.nxp.com
- HARPOON: RTOS (FreeRTOS/Zephyr) on Cortex-A using Jailhouse hypervisor:
  https://www.nxp.com/

03

Industrial Applications

Examples

Some Definitions

- Controller – A servo controller that runs algorithms to generate motion profile for motor
- IO Device – A servo driver directly connected to Permanent Magnet Synchronous Motor (PMSM) and is applying the Field-Oriented Control (FOC) technique to control the motor
- TSN Endpoint – HW that can send and receive packets complying to TSN requirements and terminating the network protocols. It can act as Controller or IO device.
- TSN Bridge – HW that can send and receive packets complying to TSN requirements and bridging the network protocols.
- AVB Endpoint – HW that can act as AVB listener and/or AVB talker and terminating the network protocols.

Distributed Motor Control Application

Distributed Motor Control via TSN network, with concurrent background traffic.

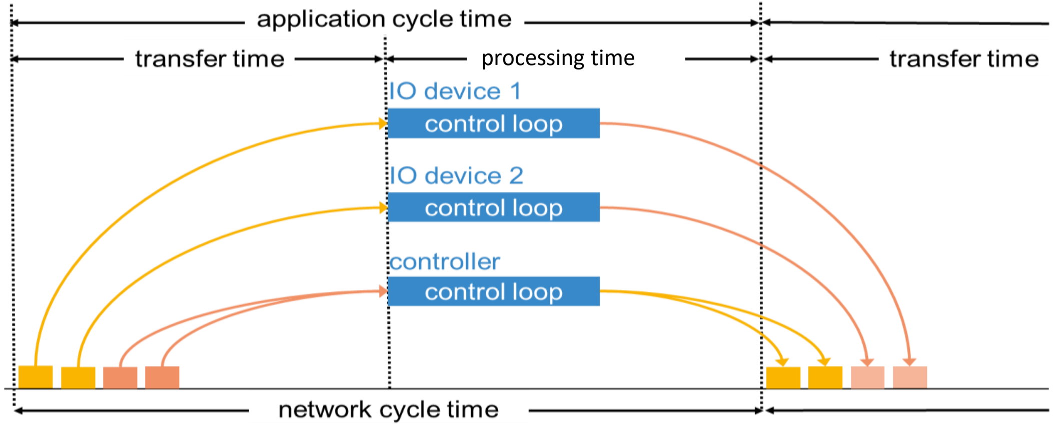
Combine i.MX RT1170 TSN endpoints with LS1028A TSN switches to form a full NXP TSN solution

IEEE 802.1AS (gPTP) network time sync, 802.1Qbv time aware shaping on i.MX RT1170 and LS1028A

Real Time processing in sync with network cycle time on i.MX endpoints

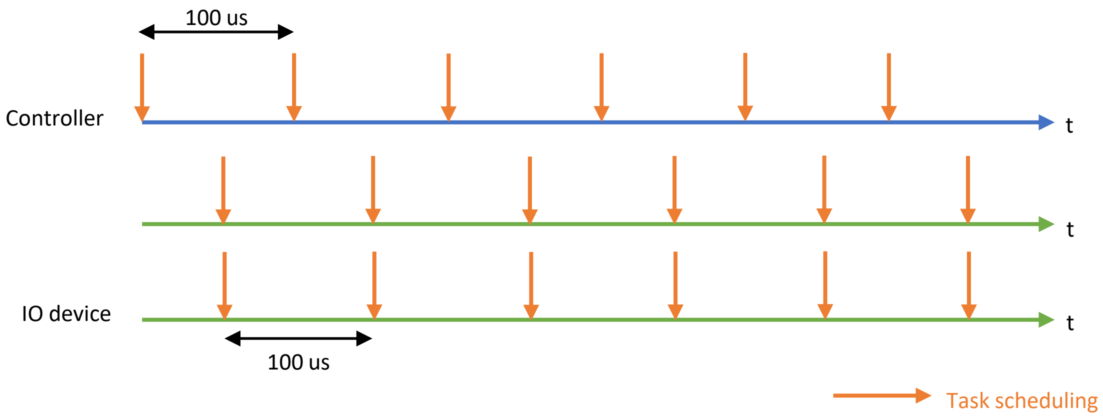
TSN Cyclic application

- The TSN example application implements a control loop similar to industrial use cases requiring cyclic isochronous exchanges over the network.
- The TSN endpoints run their application synchronized to a common time grid in the same gPTP domain so that they can send and receive network traffic in a cyclic isochronous pattern.



TSN Endpoint

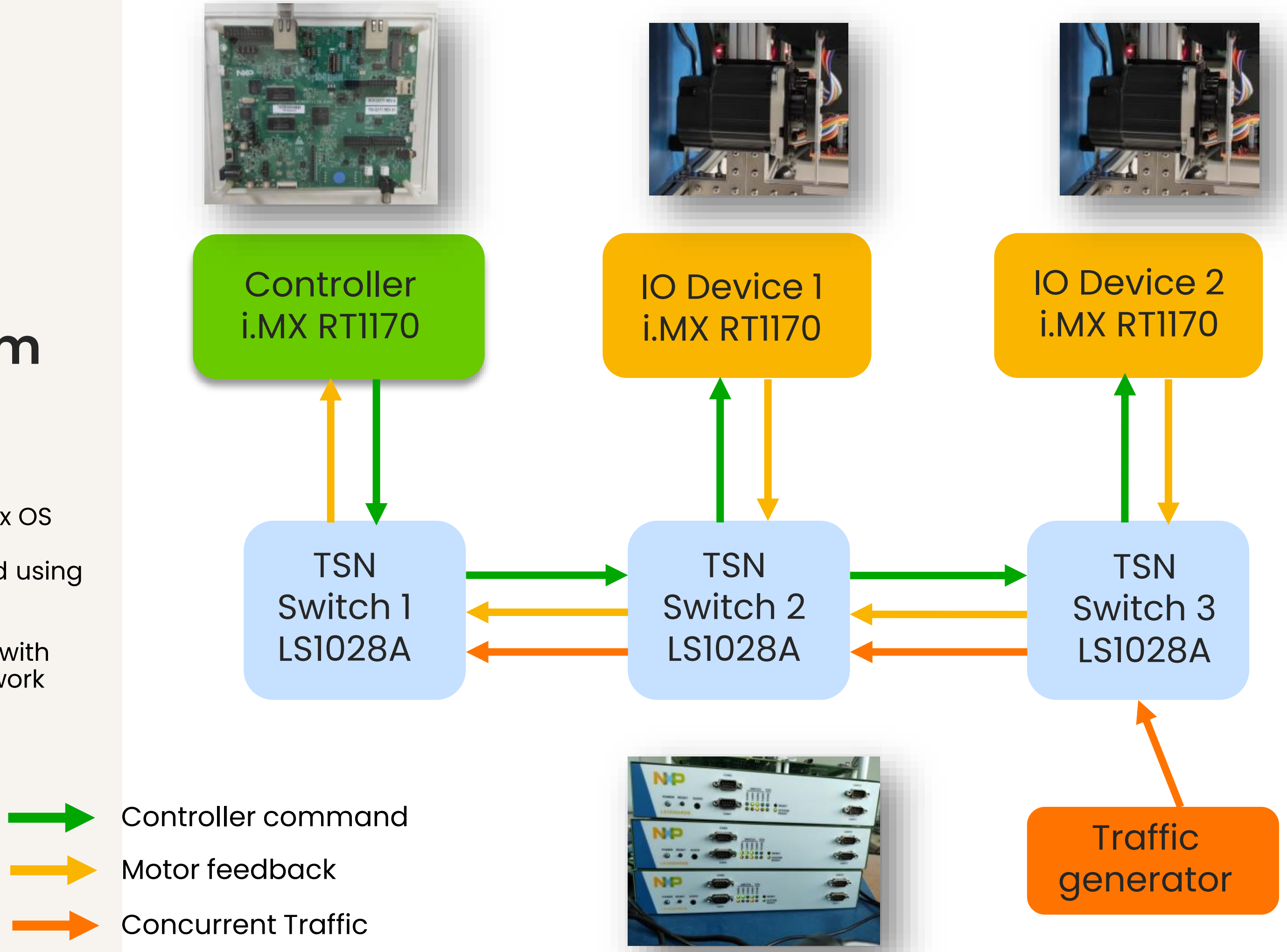
- TSN endpoints run their application synchronized to gPTP time
- 100μs period
- The controller and the IO devices are scheduled with a half cycle offset in order to reduce the processing latency.
- Time sensitive traffic is layer 2 multicast with VLAN header and proprietary EtherType
- On transmit, Enhanced Scheduled Traffic (Qbv) is used to send the time sensitive traffic at a precise time



Distributed Motor Control Application

Block Diagram

- i.MX RT1170 – Cortex-M, FreeRTOS OS
- LS1028A – Cortex-A, Linux OS
- All devices synchronized using gPTP
- Network wide schedule with Qbv enabled on all network ports



Robot machine Interface for Machine Inspection

Controlling Robotic Arm using real-time image detection and TSN technology.

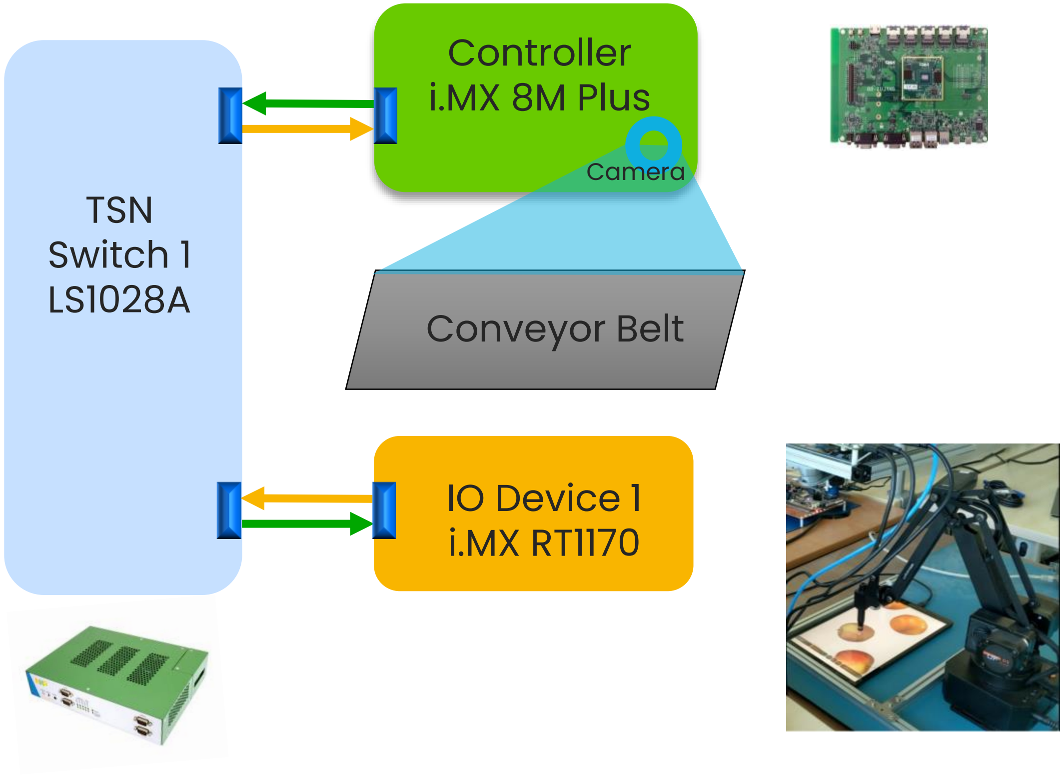
The camera connected to i.MX 8M Plus board detects the apple images and it also acts as the “Controller”. The i.MX RT1170 controlling a robot arm acts as “IO device”. The LS1028A board acts as the TSN Bridge.

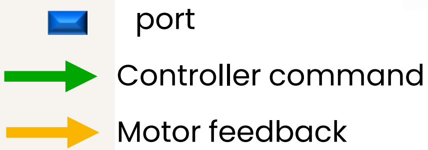
IEEE 802.1AS (gPTP) network time sync, 802.1Qbv time aware shaping on i.MX RT1170, i.MX 8M Plus and LS1028A

A small web game on smart tablet serves as a stand-in for a conveyor belt, with fruits (apples and oranges) moving on top of it.

Block Diagram

- i.MX RT1170 - Cortex-M, FreeRTOS OS
- LS1028A - Cortex-A, Linux OS
- i.MX 8M Plus – Cortex-A, Linux OS
- All devices synchronized using gPTP
- Network wide schedule with Qbv enabled on all network ports





AVB Audio Application

i.MX RT1170 platform supporting AVB Audio Listener role and connected through an AVB bridge

The i.MX 8M Plus board acts as AVB Audio Talker

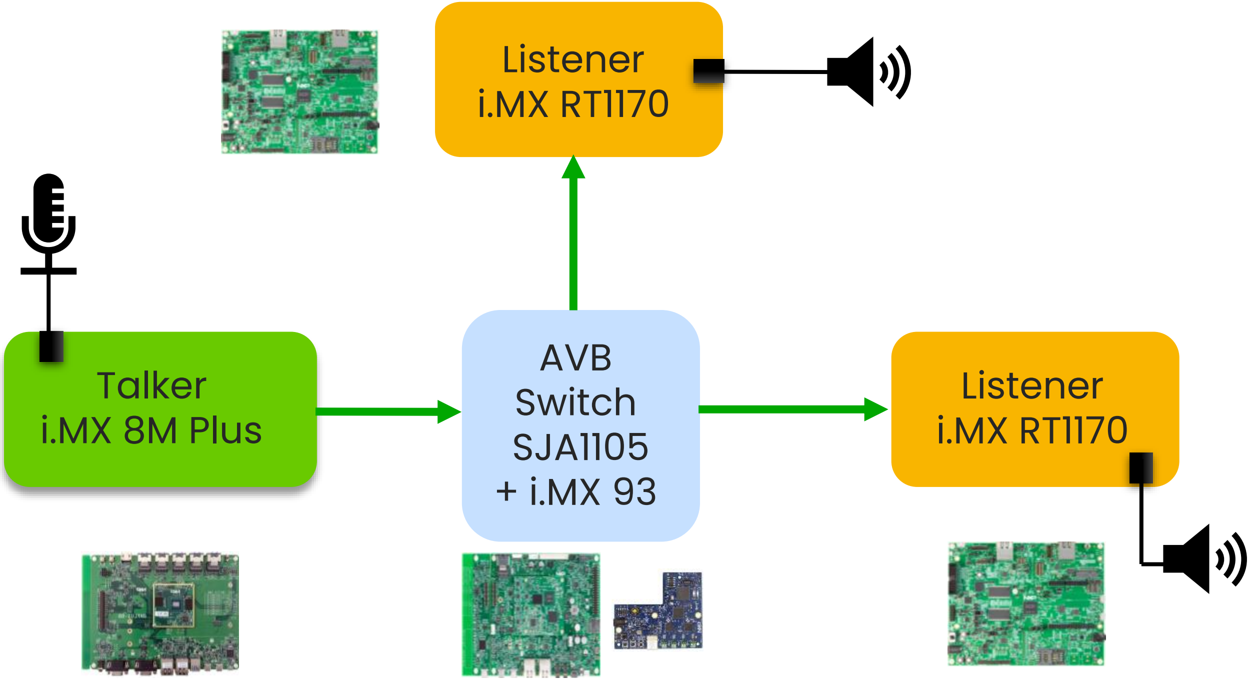
The i.MX 93 and SJA1105 combination boards act as AVB Bridge

IEEE 802.1AS (gPTP) network time sync, 802.1Qav Credit Based Shaping

Audio Real Time processing on i.MX RT endpoints with guaranteed network transport latency

Block Diagram

- i.MX 8M Plus - Cortex-A, Zephyr OS (hypervisor)
- i.MX RT1170 – Cortex-M, FreeRTOS OS
- i.MX 93 – Cortex-A, Linux OS
- All devices synchronized using gPTP
- AVDECC, SRP, AVTP protocol on all the AVB endpoints



Audio port

AVTP Stream

04

Challenges

GenAVB/TSN Stack

Some Challenges resolved in GenAVB/TSN stack

Linux OS:

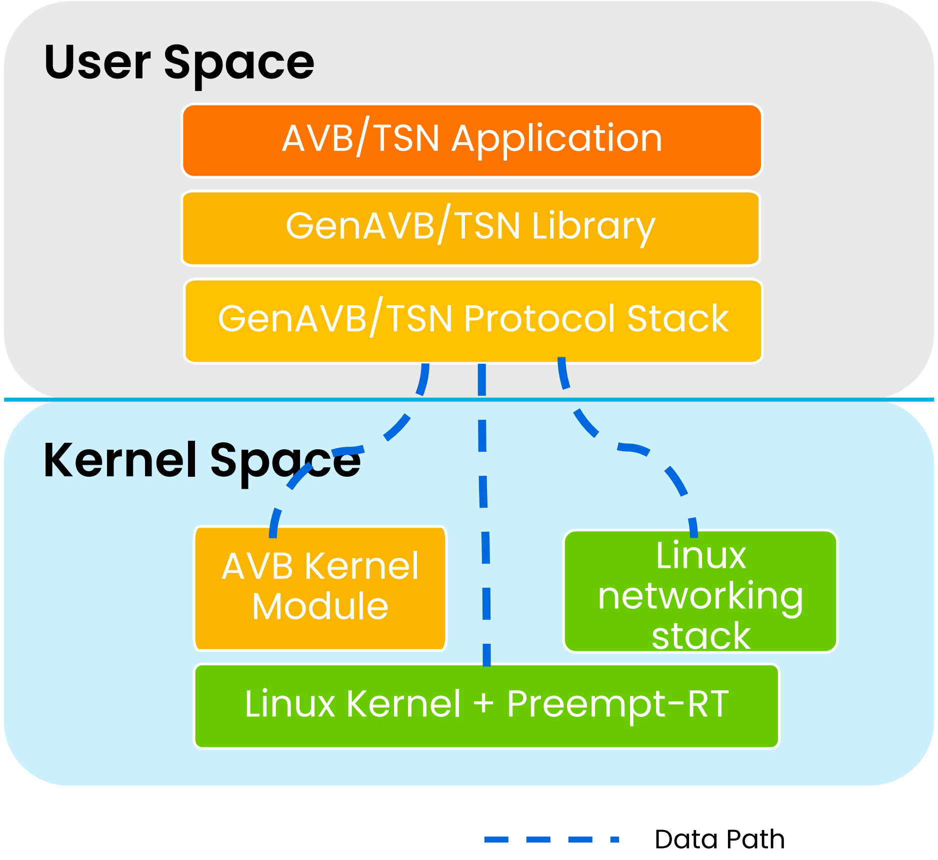
Challenge:

The overall performance of the system is constrained by real time response, and network packet processing overhead.

Our Solution:

For AVB, kernel module bypassing Linux network stack, helped improve the performance, reduced latency.

For TSN, AF_XDP sockets were used to reduce Linux network stack processing overhead.



Some Challenges resolved in GenAVB/TSN stack

Choice of OS:

Challenge:
Some AVB/TSN use cases have very low latency requirements (<10us). This target can hardly be met under Linux OS.

Our Solution:
FreeRTOS or Zephyr OS running on Cortex-A handling critical data path, and controlled by Linux OS using Jailhouse (hypervisor)

MCU support:

Challenge:

Limited memory and CPU resources in Cortex-M devices

Our Solution(s):
Careful partitioning of code and data, into specific memory regions, differentiating critical/often used code from non-critical/seldom used.
Dynamic allocation of resources at initialization time, tailored for the specific use case.
Hard limit on number of packets processed per time unit, to limit and smooth CPU load/memory utilization.

Questions

Get in touch:

https://www.nxp.com/support
https://github.com/NXP/GenAVB_TSN/issues

nxp.com

nxp.com